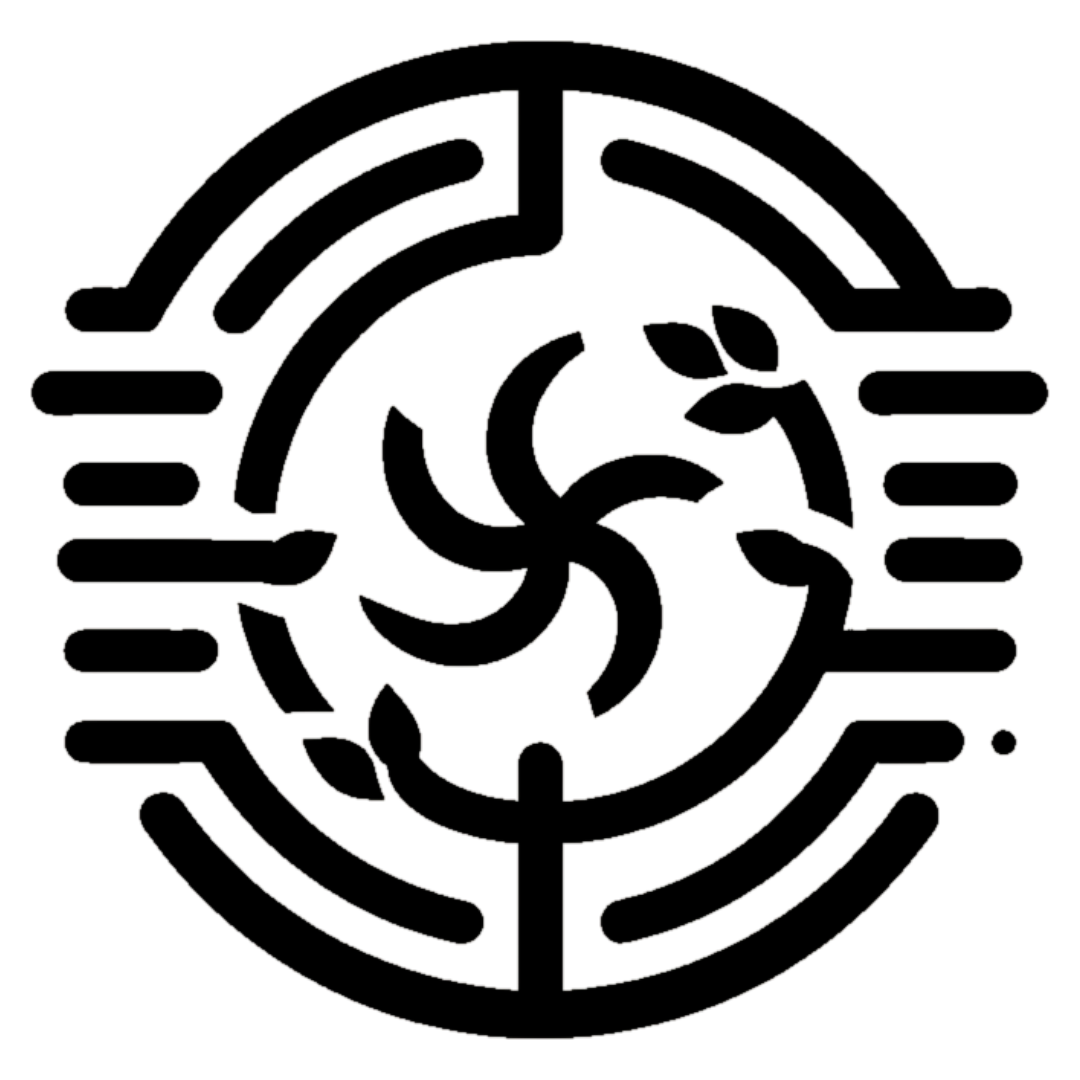
7/5/25

Brian Osmar Treviño Martínez

Sakura Inn



DOCUMENTACIÓN

Software Hotelero Sakura Inn

# INDICE

[INDICE 1](#_Toc197483414)

[INDICE DE FIGURAS 3](#_Toc197483415)

[1. INTRODUCCIÓN 4](#_Toc197483416)

[2. RESUMEN 5](#_Toc197483417)

[3. ABSTRACT 6](#_Toc197483418)

[4. REQUISITOS DEL SISTEMA 7](#_Toc197483419)

[4.1 Compatibilidad y Responsividad 7](#_Toc197483420)

[4.2 Dependencias Técnicas 7](#_Toc197483421)

[4.3 Notas Adicionales 7](#_Toc197483422)

[5. LISTADO DE TECNOLOGIAS 8](#_Toc197483423)

[6. ESTRUCTURA DEL PROYECTO 9](#_Toc197483424)

[6.1 Carpetas del proyecto 9](#_Toc197483425)

[6.2 Git 9](#_Toc197483426)

[6.3 Vscode 10](#_Toc197483427)

[6.4 Assets 10](#_Toc197483428)

[6.5 Css 11](#_Toc197483429)

[6.7 JS 12](#_Toc197483430)

[6.7 Views 13](#_Toc197483431)

[7. DIAGRAMAS DEL SOFTWARE 14](#_Toc197483432)

[7.1 Inicio 14](#_Toc197483433)

[7.1.1 Explicación de los elementos del diagrama de inicio 15](#_Toc197483434)

[7.2 Admin 20](#_Toc197483435)

[7.2.1 Explicación de los elementos del diagrama del Administrador 21](#_Toc197483436)

[7.3 Cliente 25](#_Toc197483437)

[7.3.1 Explicación de los elementos del diagrama del cliente 26](#_Toc197483438)

[7.4 Mas Información 28](#_Toc197483439)

[8. Instalación y Despliegue 29](#_Toc197483440)

[8.1 Repositorio de GitHub 29](#_Toc197483441)

[8.2 Página en Vivo (GitHub Pages) 30](#_Toc197483442)

[8.3 Instalación Local 30](#_Toc197483443)

[8.4 Requisitos Previos 31](#_Toc197483444)

[8.5 Notas Importantes 31](#_Toc197483445)

[9. Soporte 32](#_Toc197483446)

# INDICE DE FIGURAS

[Ilustración 1 Estructura de Carpetas en VS Code 9](#_Toc197483447)

[Ilustración 2 Contenido de la carpeta .git 9](#_Toc197483448)

[Ilustración 3 Contenido de la carpeta assets 10](#_Toc197483449)

[Ilustración 4 Contenido de la carpeta css 11](#_Toc197483450)

[Ilustración 5 Contenido de la carpeta JS 12](#_Toc197483451)

[Ilustración 6 Contenido de la carpeta Views 13](#_Toc197483452)

[Ilustración 7 Diagrama de inicio 14](#_Toc197483453)

[Ilustración 8 Bienvenida a la página principal 15](#_Toc197483454)

[Ilustración 9 Carrusel de Sucursales 16](#_Toc197483455)

[Ilustración 10 Habitaciones y reseñas 17](#_Toc197483456)

[Ilustración 11 Notificación de Inicio de Sesión Necesario 17](#_Toc197483457)

[Ilustración 12 Login 18](#_Toc197483458)

[Ilustración 13 Diagrama del administrador 20](#_Toc197483459)

[Ilustración 14 Resumen general 21](#_Toc197483460)

[Ilustración 15 Historial de hospedaje 22](#_Toc197483461)

[Ilustración 16 Check-in y Check-out (confirmacion) 22](#_Toc197483462)

[Ilustración 17 administración de usuarios 23](#_Toc197483463)

[Ilustración 18 Sistema de reseñas 24](#_Toc197483464)

[Ilustración 19 Diagrama del cliente 25](#_Toc197483465)

[Ilustración 20 Correo recibido y formulario de las Reservas 26](#_Toc197483466)

[Ilustración 21 Estado de la reserva 27](#_Toc197483467)

[Ilustración 22 Repositorio en GitHub 29](#_Toc197483468)

# 1. INTRODUCCIÓN

El proyecto **Sakura Inn** nace como una solución tecnológica integral para la gestión hotelera, combinando diseño web moderno con arquitecturas cloud para ofrecer una experiencia intuitiva tanto a clientes como a administradores. Desarrollado como parte de un ejercicio práctico para aplicar conocimientos en **Firebase, Bootstrap y JavaScript**, este sistema simula las operaciones de una cadena hotelera con cinco sucursales en Japón, abordando desafíos reales como:

* **Reservas en tiempo real** con validación de disponibilidad.
* **Autenticación segura** de usuarios (clientes y staff).
* **Moderación de reseñas** para garantizar retroalimentación constructiva.

**Objetivos del Proyecto**

1. **Demostrar competencias técnicas**:
   * Uso de Firebase (**Firestore, Authentication, Hosting**) Como backend.
   * Implementación de un diseño **responsive** (mobile-first) con Bootstrap.
2. **Resolver necesidades del sector**:
   * Automatización de procesos manuales (check-in/out, reservas).
   * Centralización de datos multi-sucursal en la nube.

**Audiencia**

Esta documentación está dirigida a:

* **Equipos técnicos**: Para replicar o escalar el sistema.
* **Stakeholders**: Para entender las capacidades de la plataforma.
* **Evaluadores académicos**: Como evidencia de habilidades desarrolladas.

**Estructura del Documento**

El informe se organiza en secciones que cubren desde los requisitos técnicos hasta diagramas de flujo, garantizando una comprensión holística del proyecto.

# 2. RESUMEN

Este proyecto consistió en el diseño e implementación de un sistema hotelero web escalable, utilizando **Firebase** como solución cloud para el backend, con el objetivo de gestionar reservas, autenticación de usuarios y operaciones administrativas en una cadena de hoteles ficticia llamada **Sakura Inn**, con cinco sucursales en Japón.

El sistema permite a los clientes realizar reservas interactivas mediante un **calendario dinámico** que muestra la disponibilidad de habitaciones en tiempo real, así como funcionalidades avanzadas como **check-in/check-out anticipado** y un **sistema de reseñas moderado**. Por otro lado, los administradores pueden gestionar reservas, confirmar solicitudes de los clientes y moderar reseñas para garantizar retroalimentación constructiva, con opción a negociar mejoras en la experiencia del cliente (ej: descuentos) antes de publicar comentarios.

El resultado fue una plataforma intuitiva, con paneles de administración dedicados y navegación unificada entre las cinco sucursales, demostrando la viabilidad de arquitecturas cloud para aplicaciones empresariales en el sector turístico.

# 3. ABSTRACT

This project involved the design and implementation of a **scalable web-based hotel management system** using **Firebase** as a cloud backend solution. The system was developed for a fictional hotel chain called **Sakura Inn**, comprising five branches across Japan, and focused on handling reservations, user authentication, and administrative operations.

The platform allows customers to make **interactive bookings** through a dynamic calendar that displays real-time room availability. Advanced features include **early check-in/check-out requests** and a **moderated review system**. On the administrative side, staff can manage reservations, confirm guest requests, and moderate reviews to ensure constructive feedback—with the option to negotiate service improvements (e.g., discounts) before publishing reviews, enhancing customer satisfaction.

The result is an **intuitive platform** with dedicated admin dashboards and unified navigation across all five branches, demonstrating the effectiveness of cloud architectures for business applications in the hospitality industry.

# 4. REQUISITOS DEL SISTEMA

## 4.1 Compatibilidad y Responsividad

* **Diseño Responsivo**:
  + Adaptado desde **1920x1080px** (pantallas grandes) hasta **360x640px** (móviles).
  + Incluye elementos dinámicos como **menú hamburguesa** en dispositivos pequeños.
  + *Pruebas realizadas con Live Server en Google Chrome (versión estable más reciente)*.
* **Navegadores Compatibles**:
  + **Totalmente funcional**: Chrome (versión 100+).
  + **Funcional con limitaciones menores**:
    - Microsoft Edge: Desborde visual en botones específicos (sin afectar funcionalidad).
    - Opera GX: Comportamiento similar a Edge.

## 4.2 Dependencias Técnicas

* **Frontend**:
  + Bootstrap **5.3.3** (estructura responsive y componentes UI).
  + Firebase **11.6.0** (Firestore, Authentication, Hosting).
  + EmailJS **1.0** (notificaciones por correo).
* **Entorno de Desarrollo**:
  + Visual Studio Code + Live Server (para pruebas locales).
  + GitHub (repositorio público, sin despliegue en producción).

## 4.3 Notas Adicionales

* **No se realizaron pruebas de estrés**: Proyecto en fase de desarrollo inicial.
* **Hosting**: Solo disponible en GitHub Pages (modo demostración). Para producción, se requiere configurar Firebase Hosting.
* **Optimización**: Se recomienda validar reglas de seguridad de Firestore antes de escalar a entornos reales.

# 5. LISTADO DE TECNOLOGIAS

**1. HTML (HyperText Markup Language)**  
Lenguaje de marcado utilizado para estructurar y presentar el contenido en la web. Fue la base para construir las páginas del sistema.

**2. CSS (Cascading Style Sheets)**  
Lenguaje de estilos que permite aplicar diseño y formato visual a los elementos HTML. Se utilizó para definir colores, tamaños, posiciones y animaciones en la interfaz.

**3. Bootstrap**  
Framework de diseño basado en CSS y JavaScript que facilita la creación de interfaces web responsivas. Se empleó para lograr compatibilidad móvil y estilos consistentes con menor esfuerzo.

**4. JavaScript (JS)**  
Lenguaje de programación del lado del cliente utilizado para agregar interactividad, validaciones y manejo dinámico del DOM en el sistema.

**5. Firebase**  
Plataforma de desarrollo de aplicaciones de Google que proporciona servicios backend. Se usó como base para la autenticación, base de datos y almacenamiento.

* **Firebase Auth**  
  Servicio de autenticación de usuarios. Se implementó para permitir el registro, inicio de sesión, login social (Google, GitHub, Facebook) y gestión de sesión.
* **Firebase Firestore**  
  Base de datos NoSQL en tiempo real. Se utilizó para almacenar la información de los usuarios registrados, roles y otros datos de la aplicación.

**6. EmailJS**  
Servicio externo que permite enviar correos electrónicos directamente desde el cliente sin necesidad de un backend. Se utilizó para funcionalidades como notificaciones o formularios de contacto.

**7. Imgur**   
Servicio de almacenamiento de imágenes en la nube. Fue utilizado para subir imágenes del sistema sin almacenar archivos directamente en Firebase Storage.

**8. GitHub**  
Plataforma de control de versiones basada en Git. Se usó para gestionar el historial del proyecto, colaboraciones y respaldos de código.

**9. Visual Studio Code**  
Editor de código fuente utilizado para el desarrollo del proyecto. Proporciona extensiones útiles para HTML, CSS, JS, Firebase, y control de versiones con GitHub.

# 6. ESTRUCTURA DEL PROYECTO

## 6.1 Carpetas del proyecto

Este proyecto que fue desarrollado con la herramienta Visual Studio Code, en la cual se estructuro y modularizo el orden y almacenamiento de los recursos de este proyecto desarrollado, desde el manejo de recursos visuales, hasta carpetas especializadas para el almacenamiento de las vistas (html), las interacciones (JS) y el diseño (css), en la siguiente figura, se puede apreciar la estructura misma de estas carpetas y mas adelante entraremos en detalle de cada una:

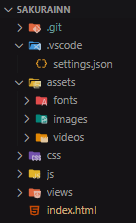


Ilustración 1 Estructura de Carpetas en VS Code

### 6.2 Git

**Propósito**:  
Esta carpeta (generalmente oculta como .git/) almacena toda la información de control de versiones de Git, incluyendo historial de commits, ramas y configuraciones del repositorio.

A continuación, en la siguiente figura se puede ver el contenido de la carpeta .git:



Ilustración 2 Contenido de la carpeta .git

**Contenido relevante:**

* **COMMIT\_EDITSG**: Gracias al uso de la extensión de GitHub en Visual Studio Code, permite hacer comentarios acerca de los cambios que se hicieron en esa versión para tenerlos en cuenta para posibles errores que se lleguen a mostrar y mantener al tanto a pesar de que pase cierto tiempo tras haber dejado el proyecto.

### 6.3 Vscode

**Propósito**:  
Contiene configuraciones específicas del editor **Visual Studio Code** para el proyecto, como extensiones recomendadas, ajustes de formato o debuggers.

**Archivos típicos**:

* settings.json: Preferencias y ajustes que se han configurado para las extensiones, en este caso para el uso de live server y previsualizar la página.

### 6.4 Assets

**Propósito**:  
Almacena **recursos estáticos multimedia** utilizados en el frontend, a continuación, vemos más a detalle su contenido:

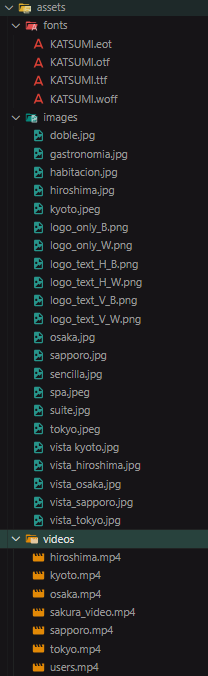


Ilustración 3 Contenido de la carpeta assets

**Subcarpetas y ejemplos**:

* **images/**:
  + logo\_only\_W.png: versión del logo con solo el icono de Sakura Inn.
  + Suite.jpg: Foto de la habitación tipo suite.
* **Videos/**:
  + Videos utilizados de fondo en los banners y el inicio de sesión de todo el sitio.
  + Kyoto.mp4: fondo del banner de la página de Kioto.
* **fonts/**:
  + Fuentes descargadas (ej.: Katsumi.woff2) para diseño web.

**Uso en HTML**:

<img src="/assets/images / logo\_only\_W.png " alt="Sakura Inn Logo">

### 6.5 Css

**Propósito**:  
Contiene **hojas de estilo** para la interfaz de usuario, incluyendo diseños responsive y componentes de Bootstrap personalizados, en este proyecto en particular se utilizó el diseño modular por plantillas citables, en lugar de definir un estilo por cada página se definieron ciertas secciones que podrán ser repetibles y fácilmente editables con este concepto, a continuación, se puede ver el contenido más a detalle:

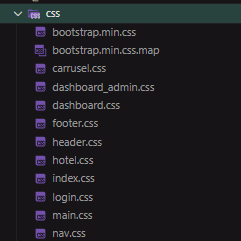


Ilustración 4 Contenido de la carpeta css

**Archivos clave**:

* hotel.css: Estilos base de todas las páginas de hoteles (colores, reset CSS).
* main.css: Estilo de fuente definido para las páginas, Katsumi títulos y Montserrat cuerpo.
* carrusel.css: Estilo del carrusel de imágenes que se presenta en la pantalla principal que redirige a los hoteles.

### 6.7 JS

**Propósito**:  
Lógica principal de la aplicación: conexión con Firebase, validaciones, y funcionalidades dinámicas, más en específico también el uso de Bootstrap para el uso y fácil manejo para un menú hamburguesa adaptable en el nav, a continuación, veremos más a detalle su contenido.

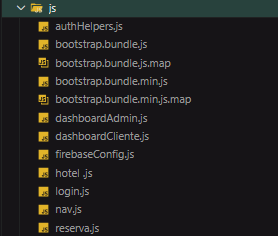


Ilustración 5 Contenido de la carpeta JS

**Archivos clave**:

* **firebase-config.js**:
  + Guardado de las credenciales de firebase y exportación a otras clases de la base de datos, app y docs.
* **reservas.js**:
  + Valida fechas seleccionadas en el calendario.
  + Envía datos a Firestore.
* **authHelpers.js**:
  + Maneja inicio de sesión con Firebase Auth.

### 6.7 Views

**Propósito**:  
Plantillas HTML de cada página del sitio índex como siempre está en la raíz, pero todas las vistas de los hoteles y los perfiles de cliente y administrador, a continuación, se puede ver más a detalle su contenido:

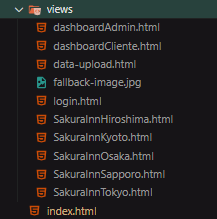


Ilustración 6 Contenido de la carpeta Views

**Estructura típica**:

* **index.html**: Página principal con descripción de la cadena hotelera.
* **dashboard.html**: Panel de control para gestionar reservas y reseñas.
* **login.html**: Pagina dedicada al inicio de sesión y registro de usuarios.

# 7. DIAGRAMAS DEL SOFTWARE

## 7.1 Inicio

En el siguiente diagrama se pueden observar distintas características, desde inicio de la página web, en la cual en el inicio se pueden observar desde. La bienvenida, servicios, sucursales, botón de inicio de sesión barra de navegación y la información de contacto, en términos generales, lo que seria el index de la web y la vista principal para todos los visitantes.

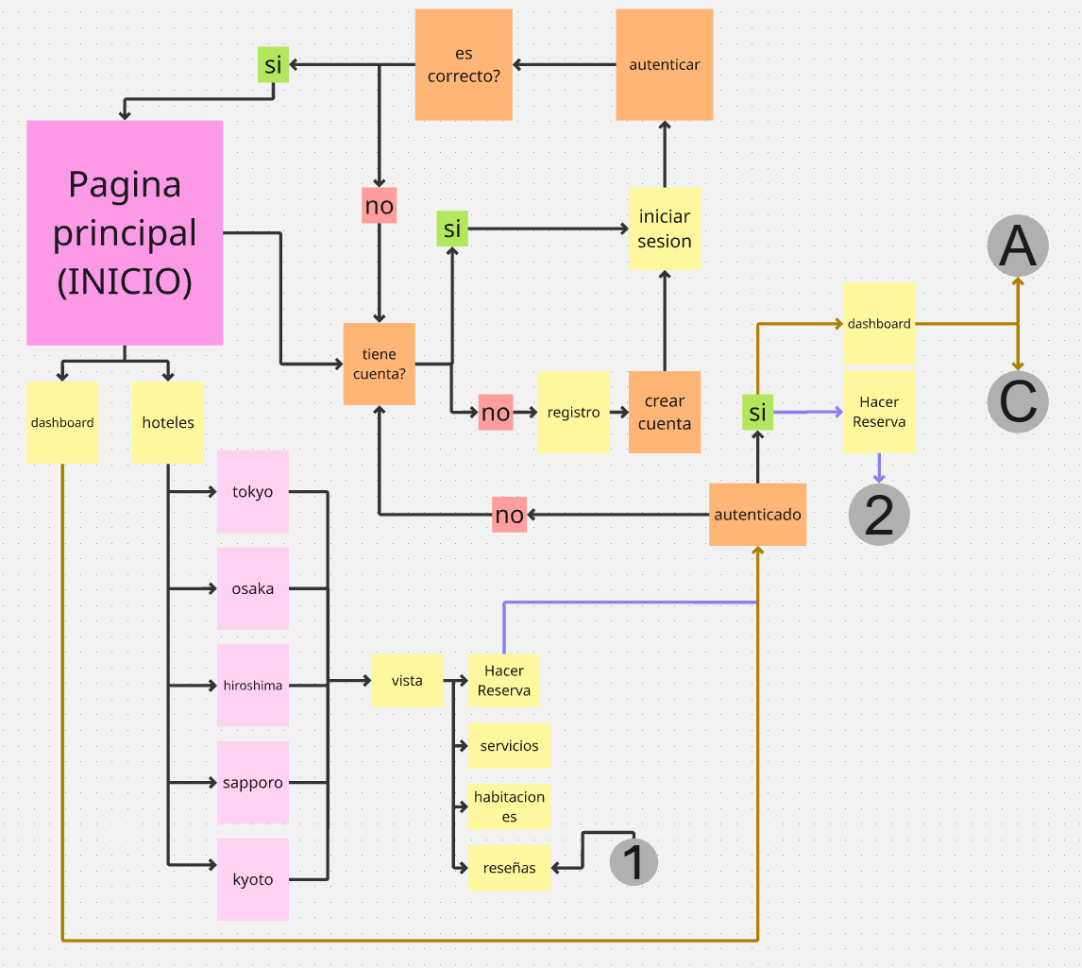


Ilustración 7 Diagrama de inicio

El diagrama inicia desde el cuadro página principal, esta se ramifica en 3 procesos.

* Dashboard(perfil)
* Hoteles
* La verificación de cuenta

### 7.1.1 Explicación de los elementos del diagrama de inicio

#### Pagina de inicio

El diagrama inicia mostrando la interfaz de la página web, se permite ver toda interfaz donde se muestra, la bienvenida y servicios, así como se ve a continuación:

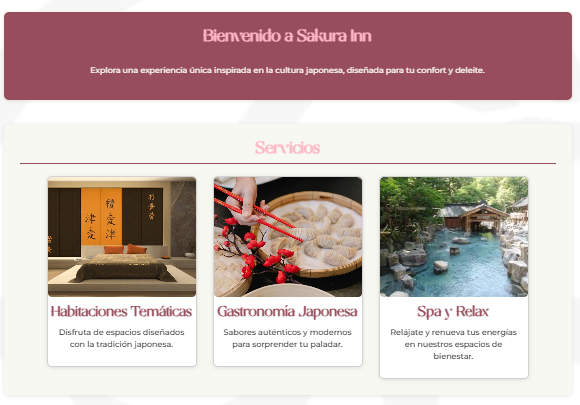


Ilustración 8 Bienvenida a la página principal

Las sucursales o hoteles que se maneja (Sakura INN), Estos hoteles abarcan ubicaciones estratégicas para abarcar una buena ubicación para que usuario tenga permitido poder viajar a ubicaciones importantes sin tener un mayor tiempo de traslado, así como se muestra en la siguiente figura:



Ilustración 9 Carrusel de Sucursales

Haciendo que el usuario tenga mayor interés en contratar los servicios de SAKURA INN.

Las sucursales que abarcan son:

* Tokio
* Osaka
* Hiroshima
* Sapporo
* Kioto

En cada sucursal se mostrará:

* Habitaciones.
* Reseñas
* Hacer reservas
* Servicios

Tal como se ve en la próxima figura:

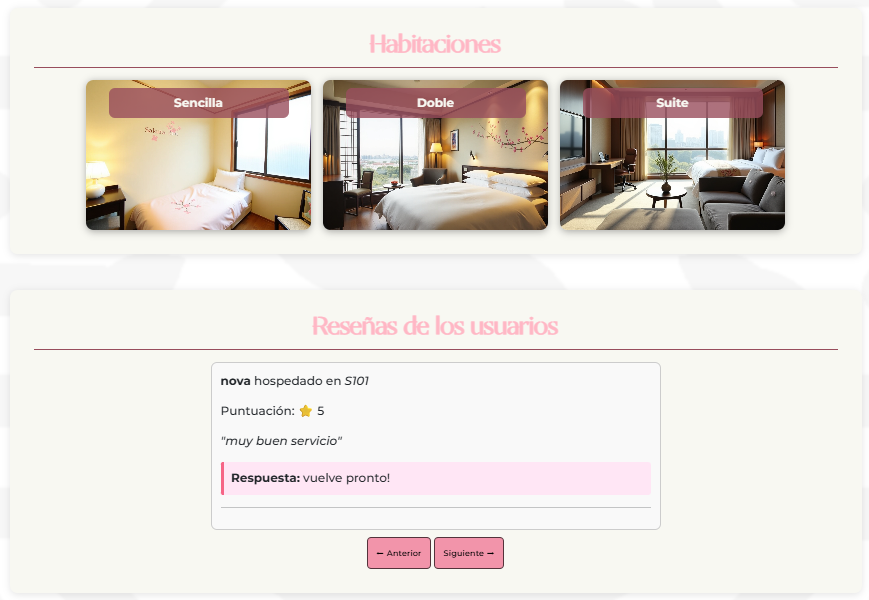


Ilustración 10 Habitaciones y reseñas

Al momento de que el usuario quiera hacer una reserva, si tiene una cuenta será redirigido al perfil de cliente para proceder a realizar su reserva, justamente una medida implementada para que solo los usuarios registrados puedan hacer una reserva:



Ilustración 11 Notificación de Inicio de Sesión Necesario

De lo contrario a no tener una cuenta, se le redireccionara al registro, donde puede agregar su correo y contraseña, dando paso a que pueda acceder a los servicios de SAKURA INN, en la próxima imagen se puede apreciar que aparte de poder iniciar sesión con correo electrónico o registrarse con este, también puede entrar con el servicio de su preferencia:

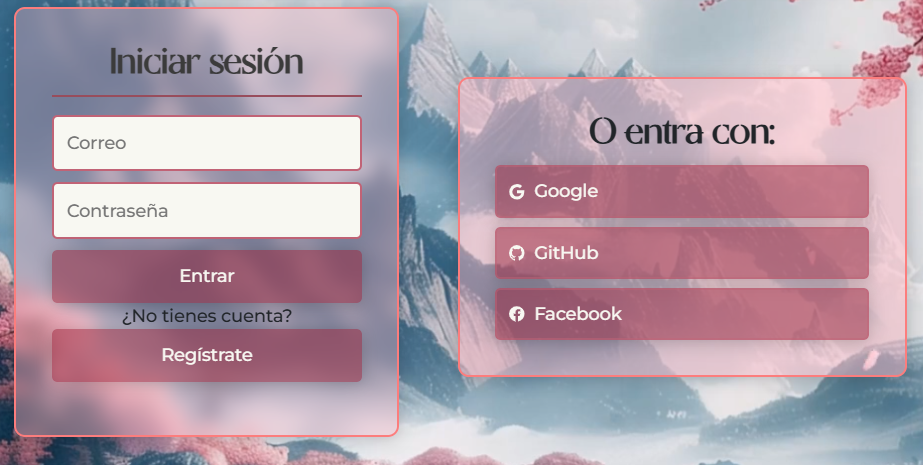


Ilustración 12 Login

#### Login

Por la parte de **registro** e **inicio de sesión** se cuenta con un ciclo de condicionales para verificar que el ambiente para el funcionamiento es el correcto, desde saber si el cliente tiene cuenta y bajo esa lógica proceder con la autenticación correcta, dentro de esta pagina que se define como **Login**, se muestran los siguientes aspectos:

* Iniciar Sesión
* Registrarse
* O entra con

Estas opciones son relevantes ya que bien la opción de iniciar sesión y registrarse se basa solo en el uso de correo electrónico, la opción de entrar con distintos servicios como Google o Facebook proveen con un mejor y eficiente acceso… y si, también al entrar con estos servicios son registrados los usuarios.

#### Roles

Finalmente, las opciones de **Dashboard** y **Hacer reserva** tienen indicadores diferentes que se aclaran mas tarde en los siguientes diagramas, ya que esto depende completamente del tipo de usuario que ha sido dado de alta en el registro y administrado posteriormente, en este caso se pueden observar 2 conectores tras la parte de autenticación que hacen referencia a la vista de los **perfiles** (Dashboards) dependiendo de que rol tengan:

* A: Administrador
* C: Cliente

Cada uno tiene ciertos privilegios sobre los que solo están visitando la pagina sin cuenta alguna en la base de datos.

## 7.2 Admin

Ahora mismo vamos a entrar en profundidad de lo que sería la “vista” del administrador dentro del sitio web, así como sus funcionalidades principales y el sistema de reseñas que fue desarrollado para esta página hotelera, a continuación, en la siguiente figura veremos como son las interacciones de esta vista:

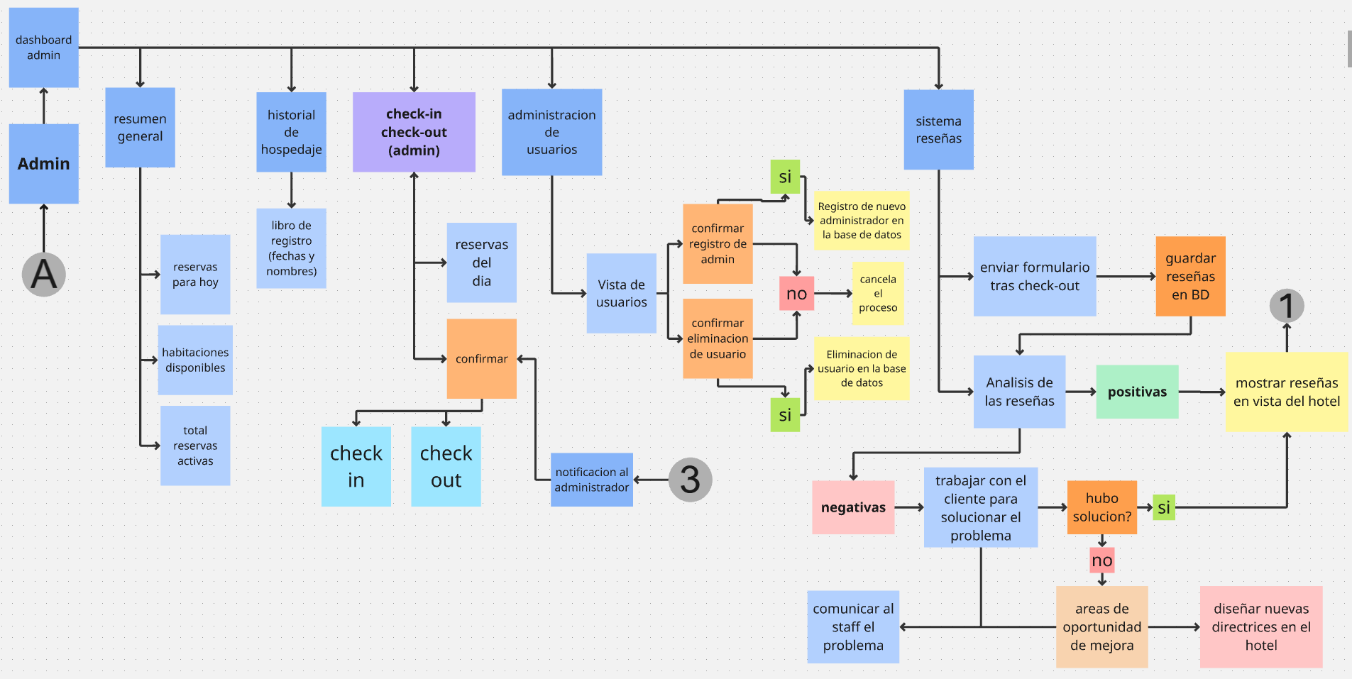


Ilustración 13 Diagrama del administrador

Como podemos observar, gracias a la conexión con el diagrama anterior (**INICIO**), tras haber hecho la autenticación o bien, inicio de sesión, si se detecta el rol de administrador podremos ver y tener estas interacciones, la vista del administrador contiene las siguientes funcionalidades:

* Resumen General
* Historial de Hospedaje
* Vista del check in y check out (perspectiva del administrador)
* Administracion de Usuarios
* Sistema de reseñas

### 7.2.1 Explicación de los elementos del diagrama del Administrador

#### Resumen General

Dentro de esta funcionalidad, apenas entremos como administradores a nuestro perfil, veremos un pequeño recuadro que nos notifica distintas cosas como se ve en la siguiente figura:

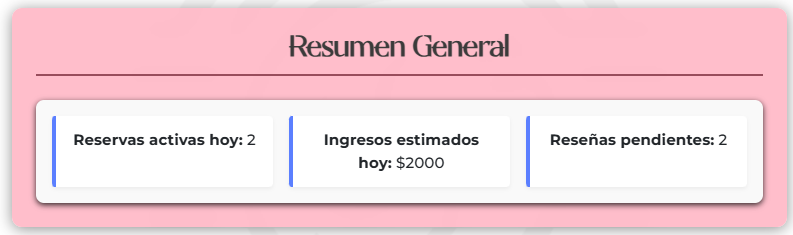


Ilustración 14 Resumen general

Tenemos a la vista lo siguiente:

* Reservas activas: el día actual cuantas reservas hay en total.
* Ingresos estimados: en base a las reservas que hay, cuanto se ganara en cuanto hay aun check-out.
* Reseñas pendientes: como tal notifica que clientes han hecho nuevas reseñas y estas requieren la aprobación del administrador.

#### Historial de Hospedaje

En esta funcionalidad, los huéspedes que con éxito el administrador les ha confirmado su Check-out, pasaran a lo que es el historial de los huéspedes que ha tenido la cadena de hoteles Sakura Inn, como se puede apreciar en la siguiente figura:



Ilustración 15 Historial de hospedaje

#### Check-in y Check out (admin)

Dentro de las posibles interacciones entre un cliente y un administrador es que el cliente, al hacer Check-in/out anticipado (avisar al hotel que está por venir o irse), es que el administrador en la recepción pueda confirmar en persona y recibir al cliente, para así, confirmar que llego con éxito al hotel y así confirmar su estadía o salida, tal y como se puede apreciar en la siguiente figura:

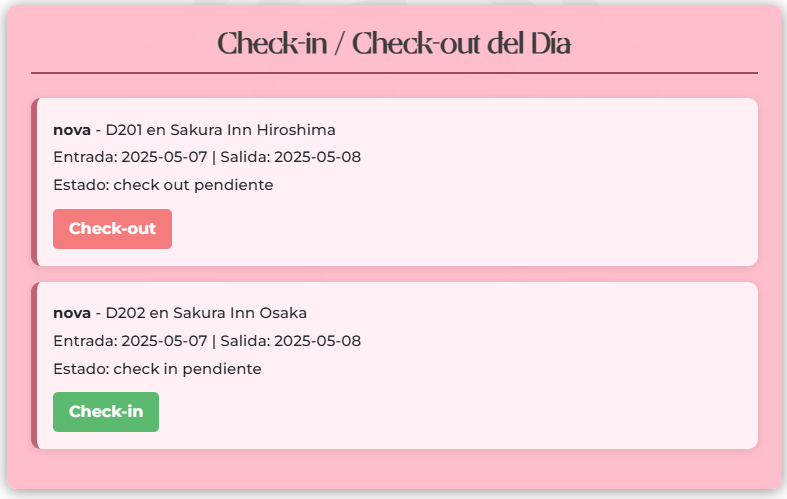


Ilustración 16 Check-in y Check-out (confirmacion)

#### administración de usuarios

Para lo que es la administración de usuarios, el administrador tiene la vista de los correos y usuarios que han sido registrados, así mismo tienen un rol asignado, tal y como se puede apreciar en la siguiente figura:



Ilustración 17 administración de usuarios

Dentro de las interacciones que podemos tener en esta funcionalidad son las siguientes:

* Rol: al apretar el botón de la columna rol, el rol del usuario cambiara al contrario (cliente a admin y viceversa), para que así se tenga la autenticación o inicio de sesión preferido.
* Acciones: en esta columna se genera el botón de eliminar, que tal como dice el nombre, se elimina al usuario de las colecciones (base de datos).

#### Sistema de Reseñas

Finalmente pero no menos importante, tenemos lo que es el sistema de reseñas, si bien se sabe que las reseñas es un derecho a la libertad de expresión, una mala reseña puede afectar al hotel, como tal esta funcionalidad no esta para limitar la voz de la gente, si no que para que los administradores actúen, al notar una mala reseña, justamente gracias a este sistema, el administrador puede hablar con el cliente para saber cual fue su problema con su estadía y así poder negociar un arreglo para mejorar este hecho, justamente en el diagrama se describe el actuar de los administradores y el staff del hotel, además de que también hay reseñas que no aportan nada bueno, ni un solo comentario constructivo y solo dan la reseña por hacerla, todas estas son vistas en cada pagina de su respectivo hotel como muestra el conector 1, pero como se ve a continuación:



Ilustración 18 Sistema de reseñas

Justamente este filtro permite ver las fallas y oportunidades de crecimiento del hotel y mejorar en base a ello, si no hay una reseña con comentarios constructivos que indiquen la falla esta puede ser rechazada, ya que una estadía en un hotel es un lujo de un servicio privado y el hotel puede reservarse para si lo que la gente quiera opinar al respecto, justamente como expone la **figura 13**, esa reseña permitirá actuar a los administrativos para solucionar el disgusto con la estadía y mejorar, de todas maneras el cliente puede dejar mas reseñas y es por eso que el administrador puede actuar para tratar de mejorar esa calificación que recibió.

## 7.3 Cliente

De la misma manera en la que podemos observar el comportamiento de un rol de administrador, los usuarios al registrarse por defecto reciben el rol de cliente y justamente a continuación veremos el diagrama que describe estas interacciones:

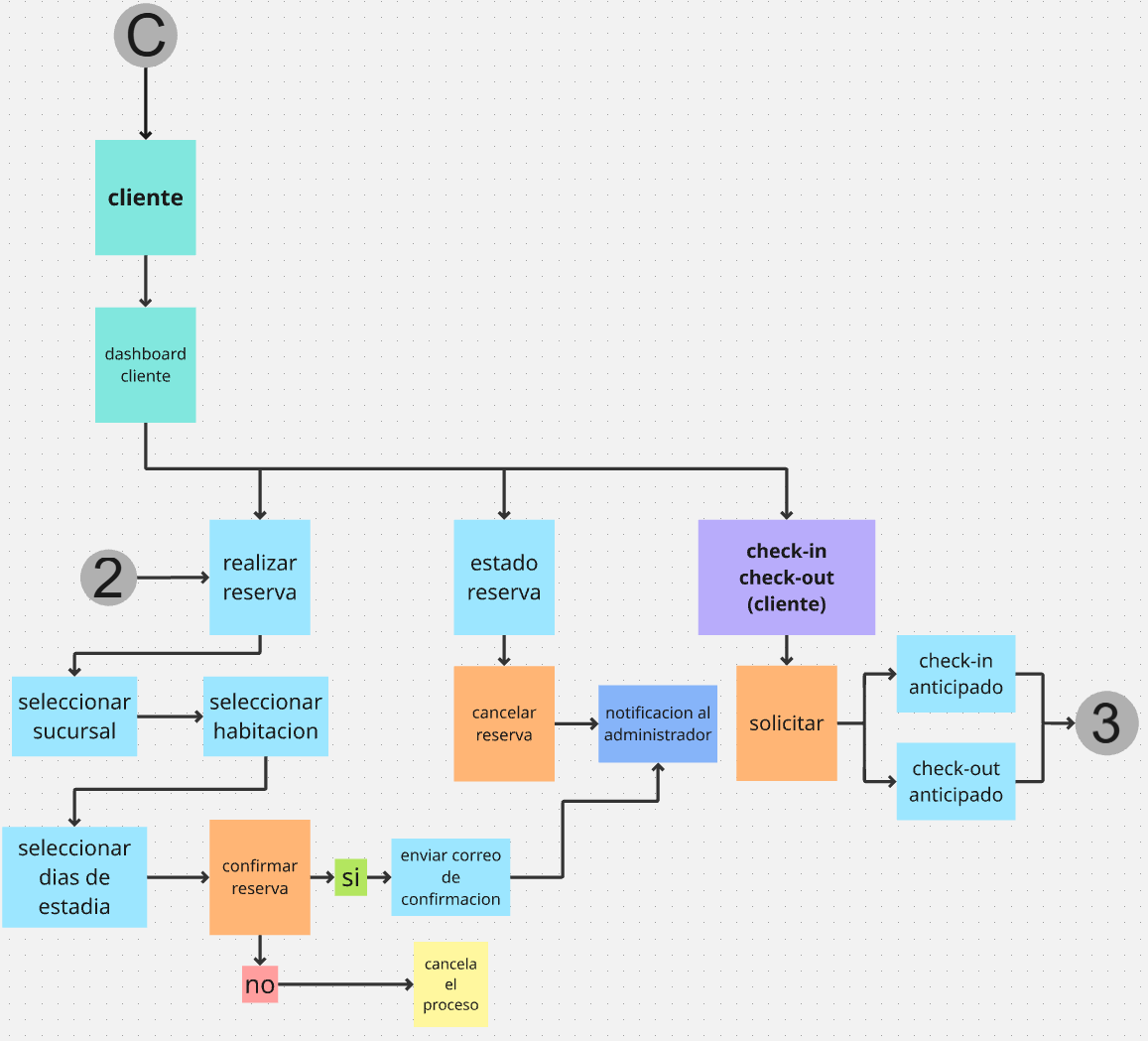


Ilustración 19 Diagrama del cliente

a primera vista se pueden apreciar las siguientes funciones, las cuales detallaremos más adelante:

* Realizar reserva
* Estado de reserva
* Check-in y Check-out (Perspectiva del cliente)

### 7.3.1 Explicación de los elementos del diagrama del cliente

#### Realizar Reserva

Dentro de este proceso que también esta conectado al proceso de la vista de los hoteles, se tiene un formulario con un calendario interactivo que permite ver al usuario una vez selecciono la sucursal y la habitación, que días esta ocupada para que lo tenga en mente a la hora de hacer una reserva y tras realizar esta, se le enviará un correo electrónico gracias a la ayuda de emailjs tal y como se puede apreciar en las siguientes figuras:

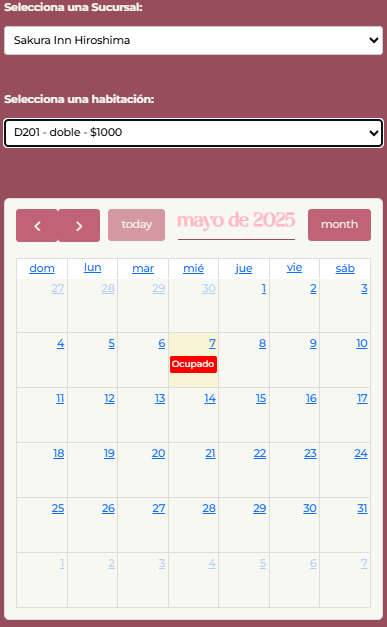


Ilustración 20 Correo recibido y formulario de las Reservas

#### Estado de Reserva

Una vez ya realizada la reserva, el cliente puede ver en la parte superior del formulario cuantas reservas tiene y detalles relacionados a ellas, como el estado actual de las reservas y sus respectivos y posibles botones de acción, como se puede ver en la siguiente figura:



Ilustración 21 Estado de la reserva

dentro de los posibles estados de reserva que tenemos son:

* **Reservado**: la reserva se acaba de realizar y el usuario puede llegar a cancelarla por su cuenta o bien hacer el check-in anticipado.
* **Check-in pendiente**: esto confirma que el cliente apretó el botón de check in y aviso al hotel de su llegada, ahora el administrador en cuanto llegue el cliente al hotel puede confirmar que está en estadía
* **En estadía**: el cliente llego con éxito al hotel y solo debe de disfrutar de los servicios que tiene disponibles, eso hasta que decida apretar el botón de check-out.
* **Check-out pendiente**: apretó el botón de check out y aviso al hotel que ya está por irse, tras esta acción en lo que el admin confirma se habilita un botón que sirve para dejar una reseña acerca de la estadía y si es aprobada por el administrador, será publicada en la página de ese hotel.

De la misma manera ahí están las interacciones de check in y check out que se hacen con el administrador para que no quede alguna duda al respecto.

## 7.4 Mas Información

Para tener una mejor visión de estas interacciones puede consultar el siguiente enlace para ir al video donde se presenta la pagina web funcional y el diseño que se le otorgo dando ese aire japón que se deseaba:

<https://youtu.be/6qQgS_tu6sg>

# 8. Instalación y Despliegue

Este apartado explica cómo clonar, configurar y ejecutar el proyecto **Sakura Inn** localmente, así como acceder a la versión desplegada en GitHub Pages.

## 8.1 Repositorio de GitHub

El código fuente del proyecto está disponible públicamente en GitHub:

<https://github.com/Novawhere/SakuraInn>

En la siguiente figura se puede apreciar como el repositorio ya está en GitHub:

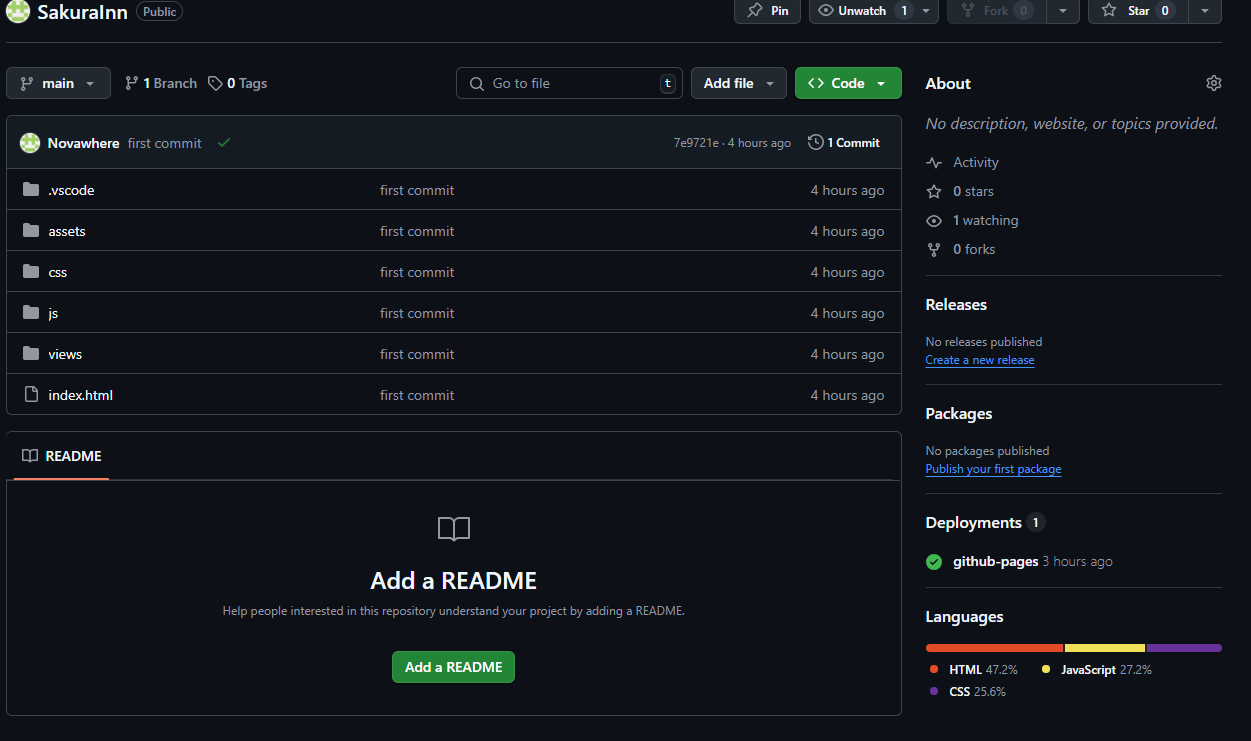


Ilustración 22 Repositorio en GitHub

**Contenido destacado**:

* Código frontend (HTML, CSS, JS).
* Configuración de Firebase (firebase-config.js).
* Assets (imágenes, íconos).

## 8.2 Página en Vivo (GitHub Pages)

Puedes explorar la versión desplegada del sitio aquí:

https://novawhere.github.io/SakuraInn/

**Nota**: Esta versión es de **demostración** y utiliza una base de datos Firebase en modo prueba.

## 8.3 Instalación Local

Sigue estos pasos para ejecutar el proyecto en tu máquina:

1. **Clonar el repositorio**:

bash

Copy

Download

git clone https://github.com/Novawhere/SakuraInn.git

cd SakuraInn

1. **Configurar Firebase**:
   * Crea un proyecto en Firebase Console.
   * Reemplaza las credenciales en /src/firebase/firebase-config.js con las de tu proyecto.
2. **Instalar dependencias** (si usas npm):

bash

Copy

Download

npm install # Solo si hay paquetes en package.json

1. **Ejecutar servidor local**:
   * Abre el archivo index.html en tu navegador o usa Live Server en VS Code.

## 8.4 Requisitos Previos

* Navegador moderno (Chrome, Firefox, Edge).
* Cuenta de Firebase (para personalizar el backend).
* Git (opcional, para control de versiones).

## 8.5 Notas Importantes

* **Firebase**: Si deseas usar tu propia base de datos, modifica las reglas de seguridad en Firestore.
* **GitHub Pages**: El despliegue es estático; las funcionalidades dinámicas requieren Firebase.
* **Personalización**: Edita config.js para cambiar nombres de sucursales o colores.

**Ejemplo de configuración de Firebase** (firebase-config.js):

javascript

Copy

Download

const firebaseConfig = {

apiKey: "TU\_API\_KEY",

authDomain: "TU\_PROYECTO.firebaseapp.com",

projectId: "TU\_PROYECTO",

};

// Inicializa Firebase

firebase.initializeApp(firebaseConfig);

# 9. Soporte

Para problemas técnicos, abre un *issue* en el repositorio:

https://novawhere.github.io/SakuraInn/